Embedded Labels for Line Features in Interactive 3D Virtual Environments

Stefan Maass*
University of Potsdam
Hasso-Plattner-Institute

Jürgen Döllner†
University of Potsdam
Hasso-Plattner-Institute

Abstract

This paper presents a novel method for labeling line features in interactive virtual 3D environments. It embeds labels into the surfaces of the annotated objects, whereas occlusion by other scene elements is minimized and overlaps between labels are resolved. Embedded labels provide a high correlation between label and annotated object – they are specifically useful in environments, where available screen-space for annotations is limited (e.g., small displays). To determine optimal positions for the annotation of line features, the degree of occlusion for each position is estimated during the real-time rendering process. We discuss a number of sampling schemes that are used to approximate the visibility measure, including an adapted variant that is particularly suitable for the integration of text based on Latin alphabets. Overlaps between embedded labels are resolved with a conflict graph, which is calculated in a preprocessing step and stores all possible overlap conflicts. To prove the applicability of our approach, we have implemented a prototype application that visualizes street names as embedded labels within a 3D virtual city model in real-time.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Virtual reality

Keywords: Labeling, Annotation, Interactive 3D Virtual Environments

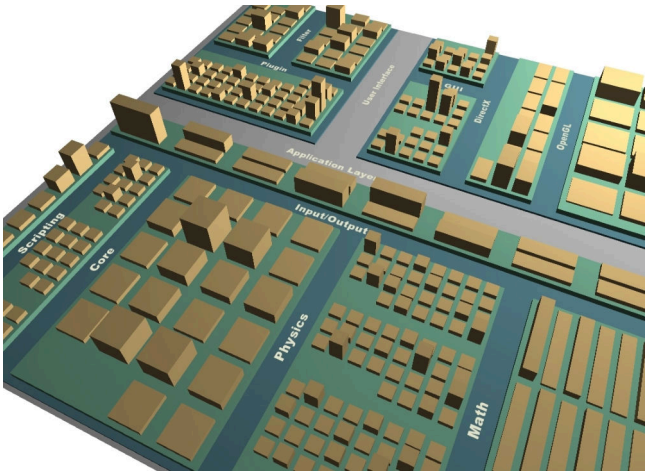

Figure 1: *3D information landscape with line features annotated by embedded labels.*

1 Introduction

The integration of text into views of 3D scenes represents a key challenge in computer graphics. It is relevant for applications and systems that need to communicate application-specific information by providing annotations to objects of 3D virtual environments. Annotations denote elements of depictions that are explicitly added as meta objects to communicate textual or symbolic information associated with objects or locations of the 3D virtual environment. Since annotation techniques generally intend to position labels closely to the annotated objects, the labels can directly and precisely encode information and, therefore, avoid additional look-ups to legends or value estimations from graphical attributes such as color gradients or texture patterns.

Typical applications and systems using annotations include medical, botanical, or engineering illustrations, diagrams, and cartographic maps. Traditionally, depictions were annotated manually in these domains, a time-consuming and error-prone design task. For maps it was estimated that up to 50% of the production time is needed for the annotation placement [Cook and Jones 1990]. As a result, especially for cartographic applications, a number of approaches for the automated annotation placement have been developed. However, only a few have been adapted to 3D model representations, cope with interactivity of digital media, or work at interactive frame rates, which are all characteristic properties of today's 3D virtual environment applications. With the increasing popularity of VR technologies the demand for such techniques increases as well. Our method contributes to reduce this gap.

The terms 'labeling' and 'annotation' are often used as synonyms. Labeling is more common in the cartographic context, due to the fact that placing names is the most common application there. Annotation is the more general term and includes the placement of symbols.

This paper presents a novel labeling technique for line features contained in 3D virtual environments. As basis for our approach, we have studied labeling techniques developed for static paper and dynamic screen maps. We adapt the concepts we found to fit characteristics of interactive 3D virtual environments, such as virtual 3D city models. For labeling at interactive frame rates we use capabilities of modern graphics hardware to find suitable positions for the placement along the feature line during frame rendering. In a second step, a conflict graph, calculated during a preprocessing phase, is used to solve overlapping conflicts between different labels.

The rest of paper is organized as follows: Section 2 discusses related work. Section 3 describes technical details of our approach including the sampling pattern used to find non-occluded positions, the position candidate selection, and the construction and evaluation of the conflict graph. Section 4 discusses the results. Section 5 draws conclusions and gives an outlook to future work.

*e-mail: stefan.maass@hpi.uni-potsdam.de
†e-mail: doellner@hpi.uni-potsdam.de

Afrigraph 2007, Grahamstown, South Africa, October 29–31, 2007.
© 2007 ACM 978-1-59593-906-7/07/0010 $5.00

2 Related Work

2.1 Cartographic Label Placement

Label placement is well studied in cartography. An excellent bibliography on this topic is provided by Wolff and Strijk [1]. Generally, cartographic maps demand for techniques to annotate point, line, and area features, such as cities, streets, and, states. A comparative overview of point feature labeling is given by the work of Christensen et al. [1995]. Most of the algorithms use discrete position candidates and select one of them during an optimization process.

Line feature labeling techniques are used to annotate rivers, contour lines, or street networks. Poon [1997] presents an aproach to maximize the label with on rectilinear maps, inclusive statements about the complexity of this problem. However, in our case we are aiming to maximize the visibility of labels instead of their extent. The work of Wolff et al. [1999] focuses on a method that aesthetically place curved labels along a poly line. Their placement criteria include a minimal and maximal distance to the line, an acceptable curvature maximum, non-intersection of the label and the poly line, minimal number of inflection points, a minimal bending, and a maximal horizontal placement.

Automated labeling in general is a complex task. Marks and Shieber [1995] show that it is NP-complete, even for simple placement problems. Most of the cartographic approaches aim to the labeling of static maps, so the processing time is subordinate compared to the placement quality. As a result, only a few cartographic approaches make use of interaction or animation. One example is given by Chigona et al. [2001], where area features morph to a rectangular shape to improve the legibility of the presented text inside. An approach for the labeling of maps that allows an interactive change of the scale and the map region visible on screen is presented by Petzold et al. [2003]. In a preprocessing step, they calculate a conflict graph that is queried during the interaction to identify possible conflicts between labels for a defined scale; an idea we adapted in our approach.

2.2 Annotation of Virtual 3D Environments

2.2.1 Annotations in Different Application Domains

For the use of annotations in 3D interactive virtual environments three major application domains exist: virtual illustrations, 3D spatial visualization, and augmented reality. In virtual illustration applications the object of interest is typically centered in the view, and the user is inspecting it by rotation and scaling operations. Thereby the region around the object is usually available as placement space. In contrast, applications in the 3D spatial visualization and augmented reality domains provide navigation metaphors related to a terrain, such as walking, driving, or flying, resulting in a more immersive perspective where unused space is rare. Furthermore, in augmented reality users might be more sensitive to misleading depth cues (e.g., occlusion), caused by imprecise placed annotations [Drascic and Milgram 1996]. The immersive and stereoscopically perceived environment increase the motivation for annotations that virtually embeds into their reference.

2.2.2 Object-Space and Screen-Space Annotations

In 3D virtual environments, annotations can be added before perspective transformation or afterwards, thus annotation techniques can be divided into object-space and screen-space techniques. The screen-space technique approach is commonly used for representations that are dominant in two-dimensions (e.g., maps) or illustrations at static media (e.g., paper). In these settings annotations are typically depicted without perspective deformation, which improves the legibility, but can reduce the correlation to the reference. For automated techniques, annotation in screen-space has two major advantages. First, the overlay of other annotations or important areas of the depictions can be easily detected, because the problem space is reduced to a 2D plane. Second, screen-space annotation techniques work independent from the content, because a usual image serves as input. This makes them easy to be implemented, usable independent from the concrete image synthesizing process and allows to handle, transmit, and store image and annotation information separately.

A first approach for the labeling of illustrations of 3D objects using a screen-space technique is presented by Preim et al. [1997]. Their technique restricts the placement to container areas around the object of interest. Annotations are placed inside these containers and are linked to the corresponding feature with an explicit line. An approach, where annotation can take place over or near their reference is developed by Bell et al. [2000; 2001]. They present a view-management data structure that allows fast query and marking operations for rectangular regions of the screen. Initialized with the screen-space bounding boxes of important scene elements, this is used to avoid an overlay of these scene elements and of annotations placed before. Another view-management method, optimized for real-time point-feature labeling on terrain-based 3D virtual environments, is described by Maass and Döllner [2006b].

As done earlier in cartography [Ebner et al. 2003], Hartmann et al. [2004] apply the concept of potential fields to the screen-space labeling of 3D illustrations. Thereby, forces between labels, their references, and the screen boundary are defined for the initial positions. After this, the placements are improved with an iterative relaxation process. Götzelmann et al. present an agent-based approach to label 3D illustrations [2006a] and techniques that focus on the contextual grouping of labels [2006b] or the annotation of animated object parts [2007].

Object-space techniques embed annotations into the virtual 3D scene, so they become part of it and are presented to the viewer with correct perspective attributes. The most natural form is to draw or write the information directly onto object surfaces. With billboarding [Akenine-Möller and Haines 2002] objects used for annotations, such as planar signs, can be aligned parallel to the view plane to force a high screen presence and legibility. Maass and Döllner [2006a] extend this principle to select annotation positions on building surfaces and orient annotations depending to the surface variation near to this place. This leads to annotations that embed like real-world signs and slide around the edges they approach.

However, between these two fundamental approaches, screen-space and object-space annotation, hybrid methods can combine advantages from both worlds: Object-space techniques can use information about their location and appearance on the screen to adjust their position in the scene for a better visibility, legibility, or correlation to the annotated object. On the other hand, screen-space techniques can distort annotations afterwards or scale them depending on the distance to simulate depth cues and thereby improve the perspective impression.

[1] http://liinwww.ira.uka.de/bibliography/Theory/map.labeling.html

3 Labeling Technique

3.1 Context and Requirements

Even across different application domains, general rules for the placement of labels exist: they should be placed legible, near the annotated object, and should not overlap other labels or important regions of the depiction. In addition, for the annotation of line features on two-dimensional maps Wolff et al. [1999] stated that:

a) Annotations should keep a minimal distance to the line but reside near to it,

b) should only bend up to a maximum, depending on the curvature of the feature line,

c) avoid to intersect the feature line or itself,

d) minimize the number of inflection points, and

e) should be placed as straight and as horizontally as possible.

Because our approach aims at the annotation of straight line features in interactive 3D virtual environments, these rules have to be adapted. Instead of placing the name above or below the line, we embed the labels into the surface of the annotated objects (Fig. 2), which maximizes the correlation between label and annotated object. The labels are allowed to slide along the line feature, so problems to be solved by the labeling technique include:

1. For each label, find potential label position candidates where the label can be embedded unoccluded by other scene objects (Fig. 3a). If there is no such placement, find positions that maximize visibility.
2. Choose a label position candidate, such that overlaps between embedded labels are minimized (Fig. 3b).

Additionally, the text of embedded labels should stay readable after perspective projection. In the same way as labels on a paper map are placed to have the best legibility for the normal orientation of the map, embedded labels in 3D virtual environments should not show up with mixed orientations.

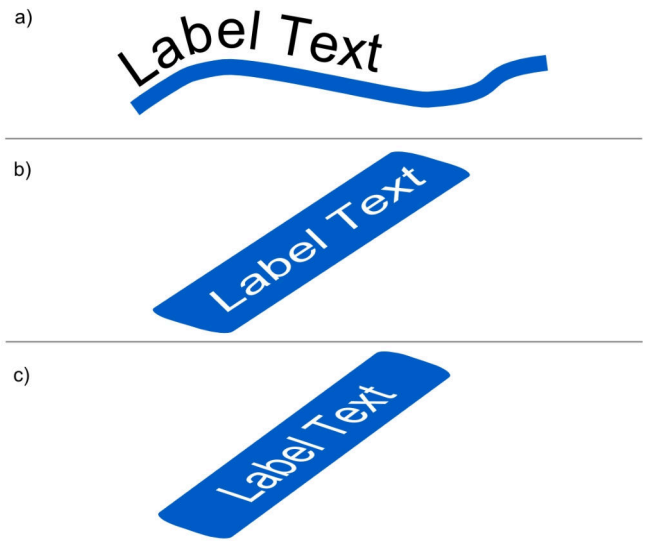

Figure 2: *Different variants of labeling a line feature: a) along the curvature on a 2D map, b) with an oriented 2D overlay on a perspective view, and c) with embedded text.*

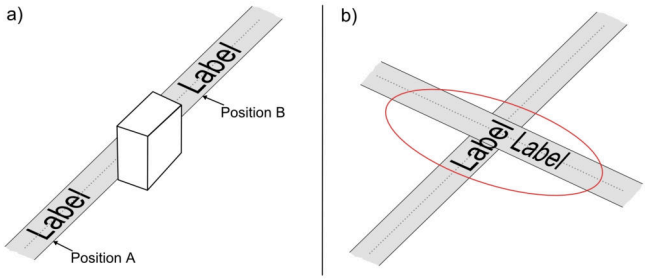

Figure 3: *For embedded labels in 3D views the annotation technique must a) find potential positions with high visibility and b) resolve overlap conflicts between labels.*

3.2 Visibility Estimation

Formally, the area of the label text A_T, that in our case has a two dimensional representation, can be written as

$$A_T = \int_{x_0}^{x_1} \int_{y_0}^{y_1} k(x,y)dydx,\ x,y \in \mathbb{R} \tag{1}$$

where $k(x,y)$ denotes a function that returns 1 if the area element is part of a letter glyph, otherwise 0. For a better understanding, but without loss of generality, in the following we assume that the x-axis is oriented parallel to the writing direction and located in middle of the text. The y-axis is orthogonal to the x-axis and points into the up-direction of the text. Conceptually, the line feature defines the x-axis of the label, so it can slide along it. This way we can define the anchor point by a normalized parameter $t \in [0..1] \subset \mathbb{R}$. For $t = 0$ the label is placed at the beginning of the line feature, for $t = 1$ it is placed at the end of the line feature. For a given line feature and a given t we can define the relative visibility of an embedded label at position t by integrating over the label area.

$$v_{rel}(t) = \frac{1}{A_T} \int_{x_0}^{x_1} \int_{y_0}^{y_1} g(x+t,y) \cdot k(x,y)dydx \tag{2}$$

The visibility function is denoted by g which returns 1 if this surface element is unoccluded with respect to the current camera perspective, otherwise 0. To get a relative measure ($v_{rel} \in [0..1]$) this value is divided by the area of the text.

For label placement at interactive frame-rates, a precise visibility calculation is too complex and is not required in practice. Hence, we use an approximated visibility $\hat{v}_{rel}$ that we calculate by sampling the rectangular bounding box, containing all letters of the label, at discrete positions.

$$\hat{v}_{rel}(t) = \frac{1}{m} \sum_{i=0}^{m-1} g(f(t,i)) \tag{3}$$

Here, m denotes the number of samples used and $f(t,i)$ stands for a function calculating the i-th sample point for a label starting at t.

We have experimented with three different distribution patterns, shown in Figure 4. First, the midpoint pattern (Fig. 4a) simply distributes the samples with equal distances in a row along the line feature. This concentrates the visibility estimation on the midpoints of the letters, so that the results can be re-used for different font sizes. Second, the zigzag pattern (Fig. 4b) includes regions of the

Figure 4: *Distribution of sampling points: a) midpoint sampling, b) zigzag sampling, c) upper half zigzag sampling.*

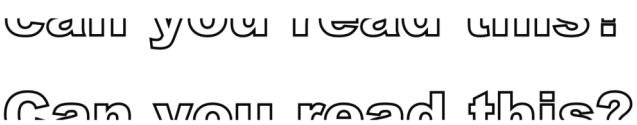

Figure 5: *Text is easier to decode for humans if the lower part instead of the upper part is missing.*

text above and below the horizontal center line, from which we assumed an increased precision for the visibility estimation. Third, a modified variant of the zigzag pattern is restricted to the upper half of the text. We expect that this is useful for label text using the Latin alphabet. The report of Bachfischer [2005] names a number of references stating that text stays rather readable if the lower part is occluded, than if the upper part is missing (Fig. 5).

In our implementation we determine the visibility for all sample points contained in the current view frustum. The technique renders the scene using a vertex and fragment shader calculating a linearized depth value for the depth buffer, relative to the near and far plane of the camera [Lapidous and Jiao 1999]. To speedup calculation and evaluation of this depth map no lighting or texturing is done in this first rendering pass, and frame buffer objects are used as render targets. Then, the precomputed sampling positions are projected accordingly to the current camera position and their depth is compared to the depth value calculated from the scene to decide if it is visible or not. The results are a stored to determine candidates with a high visibility and to select one of them later on.

3.3 Placement Position and Orientation

Sampling points and sampling results are stored ordered by their distance to one endpoint of the feature line. To identify continuous regions that are large enough to embed the label, a visibility score is calculated for each sampling position, by summing up the sampling results within the interval, defined by the length of the label. If multiple position candidates reach the maximum score, a decision for one can be made in different ways:

- *Close to the observer:* The candidate close to the observer can be chosen. Thereby, the labels are forced to positions with a large screen extent and near the point the viewer is focusing on.
- *Medium candidate:* The medium candidate can be selected as the position where the embedding starts. If there exists only one continuous unoccluded region along the reference line this will center the label within it.
- *Close to the middle of the line feature:* The sample with the highest score and the smallest distance to the midpoint of the line feature can be used. This will result in label positions favoring the middle of the line feature and avoiding border positions, which can be useful for near to top-view positions or to minimize the dynamic movement of labels during the interaction with the scene.

For a perfect embedding of a label along the feature line and into the associated surface, it can be orientated in two different ways. Both variants are needed and the selection decides whether the text appears correct or upside down after the projection. To achieve a correct appearance the endpoints of each reference line are projected to the screen. Afterwards, the selection can be easily made using relative screen positions. To avoid calculation artifacts from points behind the current viewer, the line is clipped against the current view volume before.

3.4 Resolving Conflicts

3.4.1 Conflict Graph Construction

Beside the optimization of the visibility for each label, overlaps between labels should be resolved as well. For this we adapt the idea of a conflict graph presented in [Petzold et al. 2003]. The conflict graph stores all possible overlaps between labels for a given set of line features, calculated in a preprocessing step. For the placement process a sub graph is extracted and thinned out to resolve the conflicts for the current view.

In our approach labels are embedded, therefore we do not have to store scale-dependent information in this graph. Instead, for each feature-line all possible conflicts can be represented with the following data:

- ID of the conflict feature line.
- Sample position indices that denote the placement interval for which a conflict can occur.

The IDs of conflicting lines can be easily detected by calculating crossing point between these lines within a given epsilon area. The calculation of the conflict interval is illustrated in Fig. 6. Here, the gray area marks the conflict space between two labels. If label A is placed in the interval $[t_s, t_e]$ it provokes a conflict for label B. The borders of the interval can be calculated with

$$\begin{aligned} t_s &= t_{crossing} - d_1 - d_2 - l_1, \\ t_e &= t_{crossing} + d_1 + d_2, \\ d_1 &= \frac{h_1}{tan\alpha},\ d_2 = \frac{h_2}{sin\alpha} \end{aligned} \tag{4}$$

After determining the relative interval borders, the correlating discrete sampling position indices are stored and used in the later evaluation.

3.4.2 Optimization Process

We use a heuristic to resolve occurring conflicts. For this, we sort all possible placement positions for each feature line firstly by their visibility score (number of continuous visible sample points) and secondly by their quality regarding the placement style discussed in section 3.3. We initialize the placement with a conflict sub graph that is constructed with the following steps:

- First, our algorithm removes all conflicts with labels whose score suggests a placement in regions where no conflict with other line features can appear and mark them as placed.

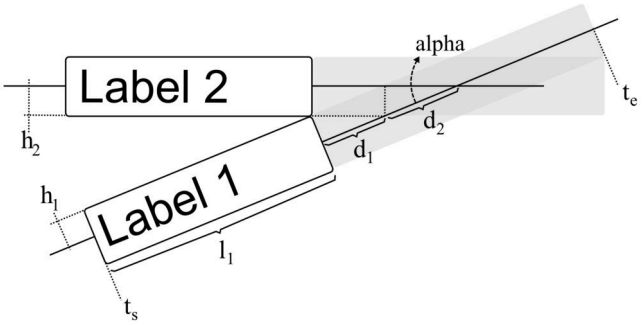

Figure 6: *Calculation of the conflict interval.*

- Because visibility is more important than the concrete position suggested by the style criteria, we mark all label placed that have another candidate with a maximum score but without blocking other labels. After that, we remove all conflicts for labels that were only blocked from ones that have been removed before. This second step is done iteratively until all direct conflicts are resolved.

This will remove most of the conflicts. The remaining labels are placed with priority given by their distance to the viewer as long as possible. This is not an optimal solution, but acceptable in interactive environments, where the viewer can quickly reach a new position that includes deselected labels. Furthermore, irresolvable conflict situations can be constructed and may appear in real-world data as well, so there has to be a strategy to handle these cases.

3.5 Dynamic Aspects

Because a small change of the view point can cause a significant change in the visibility of a reference area, labels could bounce between distant positions from frame to frame. We have implemented two techniques to smooth this effect: blending and animation. Both decouple the placement process from the interaction and start after a defined time interval without interaction. Because this time interval can be used for the placement calculations the number of labels considered or the sampling density can be increased. With blending the opacity of all visible labels is interpolated after the interaction with the scene stops, whereby they smoothly fade in. If the interaction starts again, they disappear and stay invisible until the next interaction break. With the animation technique labels are always visible. Here, labels stay pinned at their last positions as long as the user interacts with the scene. Afterwards, they smoothly slide to their new calculated position in a constant time interval.

4 Results

To evaluate our approach, we tested our technique with a virtual 3D city model, with a geometric complexity of 11642 triangles, whereas 43 line features were defined to label the streets (Fig. 7). The time for the label placement process mainly depends on the number of visible line features and the density of points used to sample the visibility. The chosen density for our model resulted in 14055 sample points, whereby only the visible ones are used during the evaluation. With these settings we reach interactive frame rates constantly between 17 and 22 fps, whereby our implementation is straight-forward, leaving enough room for further optimizations, such as the use of space partitioning schemes, adaptive sampling strategies, or multithreading. Our test system was a PC, equipped with a Pentium Core 2 Duo (2.93 GHz) processor, 2 GB memory, and a GeForce 7950 GT graphics card.

Our technique can be applied to other application domains, e.g., for 3D software visualization models that make use of a landscape metaphor [Wettel and Lanza 2007]. Fig. 1 and Fig. 8 show such a model, where our approach is used to label borders between different software modules. Compared to the midpoint sampling, only in a very few cases a qualitative improvement for the placement can be noticed using the zigzag, or the upper-half zigzag pattern. This can be traced back to the fact that in our scenarios, the embedding of labels into the surfaces combined with the perspective view transformation leads to occlusions from the bottom up to the complete letter height. Thereby, label positions calculated with the upper-half zigzag pattern longer keep a high visibility score when the viewing angle is continuously decreased.

5 Conclusions and Future Work

The presented annotation technique for line features of 3D virtual environments embeds labels into the 3D scene along their implicit reference lines. Potential label positions are determined according to a weight that takes into account label visibility and can be configured by placement styles, i.e., their distance to the observer, the midpoint within visible parts of the reference area, or the midpoint of the reference line (Fig. 8). In contrast to other annotation techniques for labeling 3D virtual environments, overlapping conflicts between labels are not detected and resolved in screen-space, but with an object-space data structure, the conflict graph. The conflict graph, created during a preprocessing step and evaluated by a heuristic placement algorithm, combined with the visibility determination in object-space provides a powerful strategy for achieving a tight and seamless integration of labels and 3D scene objects. We believe that the presented strategy can be applied and extended to other feature types as well. To increase the number of applications a level of detail (LOD) concept can be combined with our technique. This LOD concept can be used to control the visibility selection as well as the detail each label is represented by (e.g., long text, an abbreviation, or a symbol).

As a limitation, our implementation can currently only process line features. In the future, we will extend the implementation to support more complex line variants, such as networks or curved paths. This includes bending of labels along the curvature, a selection out of different segment candidates, as well as the consideration of further quality criteria during the placement, such as minimize curvature or number of inflections points. Furthermore, we will enhance the annotation technique by supporting multiple copies of annotations for the same annotated objects similar to names on long streets that are repeated multiple times to minimize the time a user needs to locate it. Despite from a top view to a 3D scene, we want to investigate if this is useful for perspective views as well, where labels are automatically placed regarding to their visibility. By an evaluation of the criteria proposed in [van Dijk et al. 2002] the quality of our approach can be analyzed. We plan to compare label placements of the discussed sampling schemes and styles against each other in more detail and in relation to an embedded labeling at fixed positions, as used in static two-dimensional paper maps.

Acknowledgements

This work has been funded by the German Federal Ministry of Education and Research (BMBF) as part of the InnoProfile research group '3D Geoinformation' (www.3dgi.de).

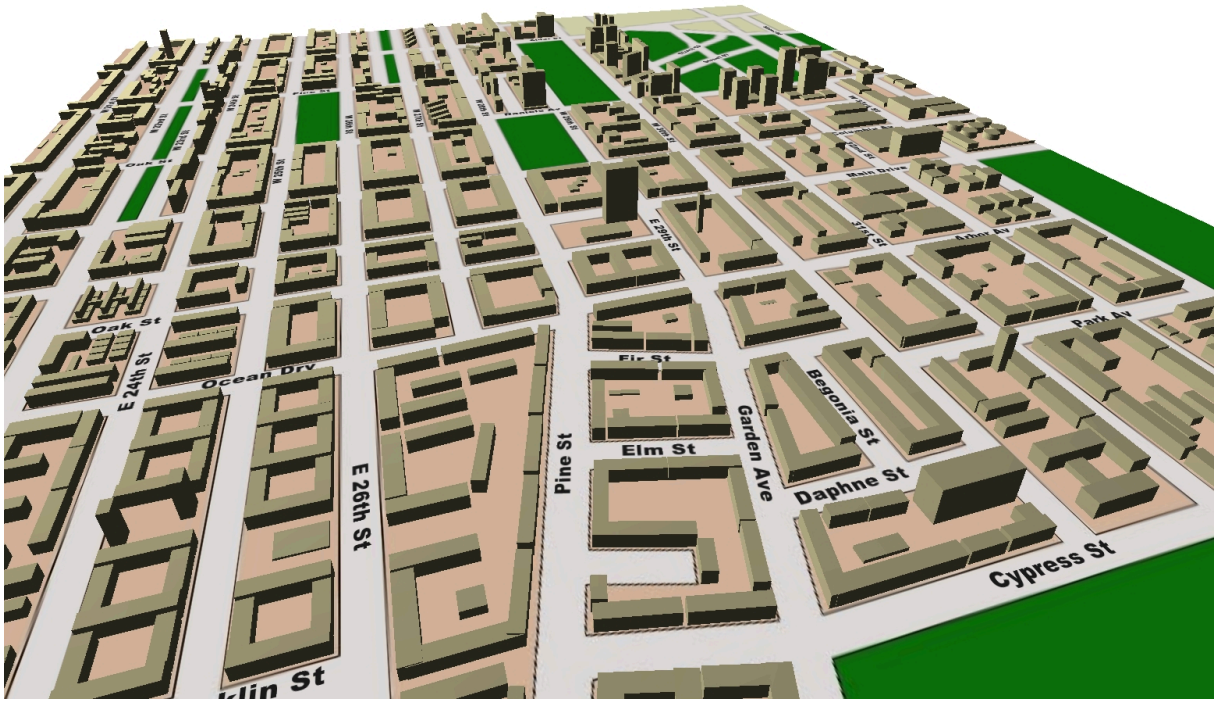

Figure 7: *Streets of a virtual 3D city model annotated by embedded labels.*

References

AKENINE-MÖLLER, T., AND HAINES, E. 2002. *Real-Time Rendering (2nd Edition)*. A K Peters, Ltd., July.

BACHFISCHER, G. 2005. Legibility and readability - a review of literature and research to understand issues referring to typography on screens and device displays. Tech. Rep. IDWoP.tech.report.05.01, University of Technology Sydney, October.

BELL, B., AND FEINER, S. 2000. Dynamic space management for user interfaces. In *Proceedings of the 13th ACM Symposium on User Interface Software and Technology (UIST)*, vol. 2 of *CHI Letters*, 239–248.

BELL, B., FEINER, S., AND HÖLLERER, T. 2001. View management for virtual and augmented reality. In *Proceedings of the 14th ACM Symposium on User Interface Software and Technology (UIST)*, ACM Press, 101–110.

CHIGONA, W., SCHLECHTWEG, S., AND STROTHOTTE, T. 2001. Dual-use of image space: The challenges of explaining visualizations from within. In *Simulation und Visualisierung 2001 (SimVis 2001), 22-23 März 2001, Magdeburg*, SCS Publishing House e.V., T. Schulze, S. Schlechtweg, and V. Hinz, Eds., 175–186.

CHRISTENSEN, J., MARKS, J., AND SHIEBER, S. 1995. An empirical study of algorithms for point-feature label placement. *ACM Transactions on Graphics 14*, 3, 203–232.

COOK, A. C., AND JONES, C. B. 1990. A prolog rule-based system for cartographic name placement. *Computer Graphics Forum 9*, 2, 109–126.

DRASCIC, D., AND MILGRAM, P. 1996. Perceptual issues in augmented reality. In *Stereoscopic Displays and Virtual Reality Systems III*, SPIE, M. T. Bolas, S. S. Fisher, and J. O. Merritt, Eds., vol. 2653, 123–134.

EBNER, D., KLAU, G. W., AND WEISKIRCHER, R. 2003. Force-based label number maximization. Tech. Rep. TR-186-1-03-02, Institut für Computergraphik und Algorithmen, Technische Universität Wien, June.

GÖTZELMANN, T., HARTMANN, K., AND STROTHOTTE, T. 2006. Agent-based annotation of interactive 3d visualizations. In *6th Int. Symposium on Smart Graphics*, Springer Verlag, Vancouver, Canada, A. Butz, B. Fischer, A. Krüger, and P. Oliver, Eds., Lecture Notes in Computer Science 4073, 24–35.

GÖTZELMANN, T., HARTMANN, K., AND STROTHOTTE, T. 2006. Contextual grouping of labels. In *SimVis*, SCS Publishing House e.V., T. Schulze, G. Horton, B. Preim, and S. Schlechtweg, Eds., 245–258.

GÖTZELMANN, T., HARTMANN, K., AND STROTHOTTE, T. 2007. Annotation of animated 3d objects. In *SimVis*, SCS Publishing House e.V., T. Schulze, B. Preim, and H. Schumann, Eds., 209–222.

HARTMANN, K., ALI, K., AND STROTHOTTE, T. 2004. Floating labels: Applying dynamic potential fields for label layout. In *4th International Symposium on Smart Graphics*, Springer-Verlag, A. Butz, A. Krüger, and P. Olivier, Eds., vol. 3031 of *Lecture Notes in Computer Science*, 101–113.

LAPIDOUS, E., AND JIAO, G. 1999. Optimal depth buffer for low-cost graphics hardware. In *HWWS '99: Proceedings of the ACM SIGGRAPH/EUROGRAPHICS workshop on Graphics hardware*, ACM Press, New York, NY, USA, 67–73.

MAASS, S., AND DÖLLNER, J. 2006. Dynamic annotation of interactive environments using object-integrated billboards. In *14-th International Conference in Central Europe on Computer*

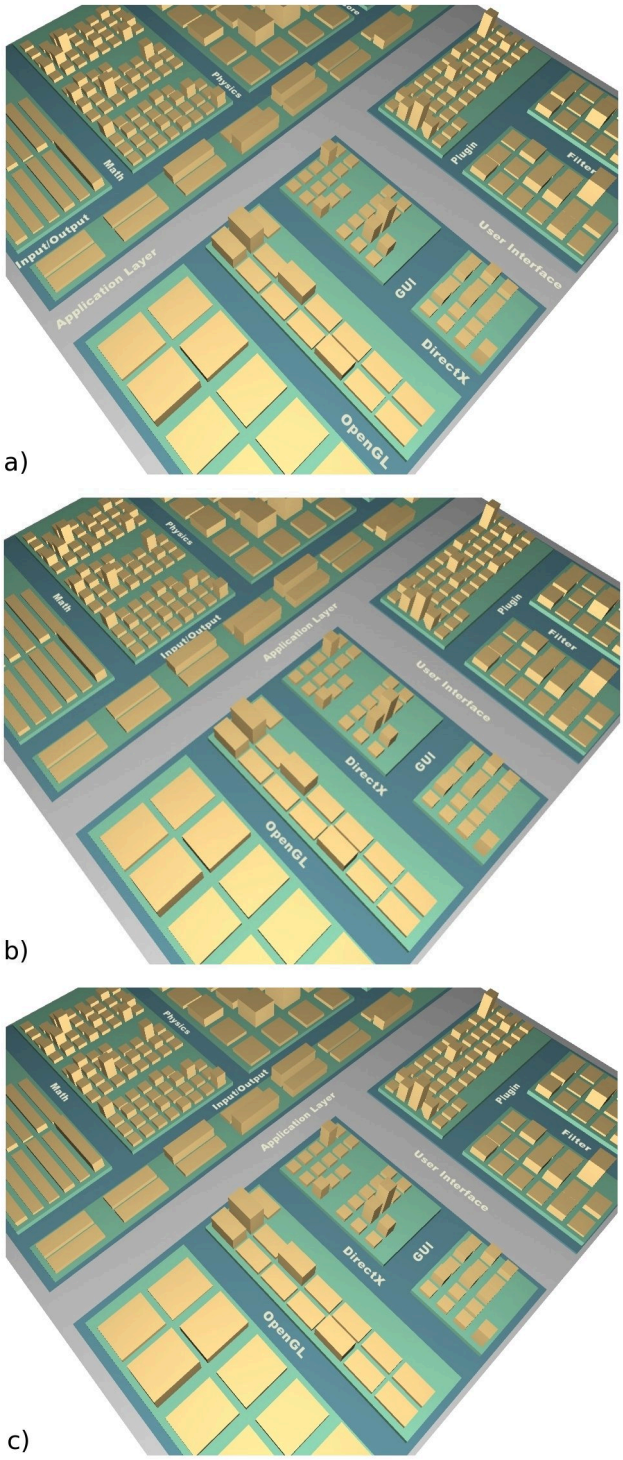

Figure 8: *Labeling of a 3D information landscape with different position selection strategies: a) close to the observer, b) medium candidate, c) close to the middle of the line feature.*

Graphics, Visualization and Computer Vision, WSCG'2006, J. Jorge and V. Skala, Eds., 327–334.

MAASS, S., AND DÖLLNER, J. 2006. Efficient view management for dynamic annotation placement in virtual landscapes. In *6th Int. Symposium on Smart Graphics*, Springer Verlag, Vancouver, Canada, A. Butz, B. Fischer, A. Krüger, and P. Oliver, Eds., Lecture Notes in Computer Science 4073, 1–12.

PETZOLD, I., GRÖGER, G., AND PLÜMER, L. 2003. Fast screen map labeling - data-structures and algorithms. In *Proc. 23rd International Cartographic Conference (ICC'03)*, 288–298.

POON, C. K., ZHU, B., AND CHIN, F. 1997. A polynomial time solution for labeling a rectilinear map. In *SCG '97: Proceedings of the thirteenth annual symposium on Computational geometry*, ACM Press, New York, NY, USA, 451–453.

PREIM, B., RAAB, A., AND STROTHOTTE, T. 1997. Coherent zooming of illustrations with 3d-graphics and text. In *Proceedings of the conference on Graphics interface '97*, Canadian Information Processing Society, Toronto, Ont., Canada, 105–113.

VAN DIJK, S., VAN KREVELD, M., STRIJK, T., AND WOLFF, A. 2002. Towards an evaluation of quality for names placement methods. *International Journal of Geographical Information Systems*.

WETTEL, R., AND LANZA, M. 2007. Visualizing software systems as cities. In *Proc. of the 4th IEEE International Workshop on Visualizing Software for Understanding and Analysis*, 92–99.

WOLFF, A., KNIPPING, L., VAN KREVELD, M., STRIJK, T., AND AGARWAL, P. K. 1999. A simple and efficient algorithm for high-quality line labeling. In *Proc. GIS Research UK 7th Annual Conference (GISRUK'99)*, D. Martin and F. Wu, Eds., 146–150.